UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/215,788	12/21/1998	JERRIE L. COFFMAN	21936435X00	1611

7590 01/31/2002

HUNG H. BULL
ANTONELLI, TERRY, STOUT & KRAUS
1300 N. 17TH STREET
SUITE 1800
ARLINGTON, VA 22209

EXAMINER	
PRIETO, BEATRIZ	
ART UNIT	PAPER NUMBER
2152	

DATE MAILED: 01/31/2002

Please find below and/or attached an Office communication concerning this application or proceeding.

H·G

Office Action Summary	Application No. 09/215,788	Applicant(s) COFFMAN ET AL.	
	Examiner B. PRIETO	Art Unit 2152	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
- Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *13 November 2001* .

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-28* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-28* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

11)☐ The proposed drawing correction filed on _____ is: a)☐ approved b)☐ disapproved by the Examiner.

If approved, corrected drawings are required in reply to this Office action.

12)☐ The oath or declaration is objected to by the Examiner.

Priority under 35 U.S.C. §§ 119 and 120

13)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____ .

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

14)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. § 119(e) (to a provisional application).

a) ☐ The translation of the foreign language provisional application has been received.

15)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. §§ 120 and/or 121.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449) Paper No(s) _____ .
4) ☐ Interview Summary (PTO-413) Paper No(s). _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: .

DETAILED ACTION

1. This communication is in response to Appeal Brief, filed 11/13/01, claims 1-28 remain pending.

2. Applicant's request for reconsideration of the finality of the rejection of the last Office action is persuasive and, therefore, the finality of that action is withdrawn.

3. Quotation of 35 U.S.C. §103(a) which forms the basis for all obviousness rejections set forth in this Office action may be found in previous office action;

4. Claims 1-18 are rejected under 35 U.S.C. 103(a) as being unpatentable over Heil et. al. (Heil) U.S. Patent No. 6,173,374 in view of Intelligent I/O (I2O) Architecture Specification (Specs), version 1.5, March 1997.

Regarding claims 1, and 14. Heil teaches a host driver configuration of a host driver module of a computer node on a server cluster, comprising:

an input/output platform (IOP) arranged to control an array of local storage devices;

Heil teaches a host driver configuration of a host driver module (col 10/line 28-50) of a computer node on a server cluster, (Heil, abstract), teaching HBA(s) embedded intelligent I_2O standard software, partitions the device driver into one module creating stackable drivers 210-280, col 10/lines 28-31, and col 10/lines 43-col 11/line 4; Software drivers 240, 250, and 260 are arranged to control an array of local storage devices, Fig. 2, wherein I/O redirector driver software 240 determines whether to satisfy a block I/O request locally or remotely, and coordinates the retrieval of data over a cluster with logically shared disks, clusters drives, comprising local stored data blocks, col 11/lines 5-11; further using software drivers ISM 250/HDM 260 to control local disks 118, col 11/lines 12-27.

a system driver module comprising: a Local Transport arranged to provide an interface to said input/output platform (IOP);

Heil teaches software driver 250 arranges to provides an interface layer to

said host (I_2O standard software) driver module structure, Fig. 2; driver (ISM) 250 is arranged to control local disks 118, to retrieve and request block level I/O requests from the local disks 118 and post blocks of data to the local disks 118, 123, col 11/lines 7-12, 28-35.

a Remote Transport arranged to provide an interface to export local storage device access onto a computer network; a host driver module architecture on a server cluster for providing I/O access storage device data transfer between computer node(s) on a server cluster system and another system;

Heil teaches software drivers 240 is arranged to provide an interface layer to a computer network; Fig. 2; provides managing means to allow the host driver module (HBA) to determine whether to satisfy a block I/O request locally or remotely, managing means coordinates the retrieval of data over a cluster with logically shared disks, cluster drives, comprising local or remote stored data blocks, col 11/lines 5-11; wherein software drivers I/O shipping 270 and HDM 280 provide an interface layer for exporting local storage device onto a computer network, I/O shipping 270 managing the reception of remotely generated requests from the other system network for local data to be exported onto the computer network, and I/O shipping HDM 280 manages the communication medium (FC) transactions in support of shipping functions, col 11/lines 36-46.

a Connection Manager arranged to establish connection services with remote servers on said computer network and coordinate functions responsible for creating a direct call path between the software driver 250 and software driver 240 to provide access to the local storage devices.

Heil teaches establishes a communication over an FC backbone network between the initial host system on a computer node and the remote HBA host system on a computer node attached to the storage location owning the requested I/O blocks, and ships the block I/O request to the peer HBA's remote disk, col 11/lines 45-65, and col 4/lines 58-61, data is transferred across computer nodes of a server cluster system by providing for I/O shipping of data blocks to peer host system, see abstract;

however Heil does not teach functions responsible for creating a direct call path between the transport driver modules to provide access to the local storage devices; nor wherein that embedded intelligent I_2O standard software comprises an IOP platform;

Specs teaches a message-based interfaces enable direct messages passing between any two device driver module for a particular class of I/O messages class, section 1.1.2.2, page 1-4, utility message (call) functions, table page 6-4 comprising message codes for accessing devices.

Upon initialization the host then gives each IOP a list of all IOPs and the physical location of their inbound message queue. When an IOP wants to connect to another IOP, it sends the request to the respective IOP's inbound message queue location. The connection request and its reply convey information enabling the two IOPs to establish a direct path for exchanging messages, section 2.1.4.1, and page 2-11.

In accordance with I_2O standard architecture, this protocol comprises a single host entity and an intelligent I/O subsystem containing a number of I/O processors entities, host comprising application(s) and their resources, executing an operating system, Fig. 2-1, an IOP platform consists of a processor, memory and I/O devices, as per architecture's standard on page 2-1.

Therefore Heil teachings wherein an HBA executing embedded intelligent software is arranged to control an array of local storage devices, inherently teach that an input/output platform (IOP) arranged to control an array of local storage devices.

It would have been obvious to one ordinary skilled in the art at the time the invention was made to enable means for creating a direct call path between the driver modules to provide access to the local storage device, as taught by Specs, motivation would be enable a direct message passing between software driver layer for a particular class of I/O, such as those further specified by I_20 standards such as LAN ports, Ethernet or Token ring controllers, SCSI ports, etc providing an architecture that is operating-vendor-independent and adapts to existing system, creating scalable drives from high-end workstations to high-end server, as taught by Specs.

Regarding claim 15, IOP supports at least one or more input/output processors (See Specs: I2O standard hardware architecture is defined for a single host entity and an intelligent I/O subsystem containing a number of I/O processors entities, host comprising application(s) and their resources, executing an operating system, Fig. 2-1, an IOP platform consists of a processor, memory and I/O devices, page 2-1), and comprises:

a device driver module which interfaces the local storage devices for controlling said array of local storage devices (Heil, col 11/lines 12-35, device driver module 260); and

a communication layer which defines a mechanism for communications between the system driver module and the device driver module (see Specs: I_2O communication layer (messaging service layer), which delivers I/O transactions messages from one software module to another, any where in the I_2O domain, independent of the modules hierarchy, section 2.1.3, page 2-4, Fig. 2-3).

Regarding claim 2, IOP access module is one of a hardware module (Heil: col 11/lines 28-35), a combined hardware/software module (Heil: col 10/lines 43-50), and a software module provided on a tangible medium (Heil: col 10/lines 28-50).

Regarding claims 3 and 8, discussed on claims claim 1 and 14, further wherein a communication between host computer nodes on a server cluster system and another system (Heil. host computer nodes on a server cluster, abstract, system interconnected via a data network, col 9/lines 11-17).

Regarding claim 4, said IOP which comprises: at least one or more input/output processors (see Specs, page 2-1); at least one storage device as said input/output devices (Heil: disks 118, col 11/lines 7-35); a device driver module arranged to control access via interface means with said storage device (Heil: software driver 260, col 11/lines 7-12, 28-35); a communication layer which defines a mechanism for communications between the ISM driver 250 (Local Transport) and the HDM (260); (Heil: software driver 250, col 11/lines 12-27, supporting communication between 250 and 260, see Specs messaging (communication) layer section 2.1.3, page 2-4, Fig. 2-3).

Regarding claims 5, 6, 9, 10, and 16, wherein said communication layer is responsible for managing and dispatching all service requests (see Specs: section 2.1.3, page 2-4, Fig. 2-3 communication layer is a network of object instance) and providing a set of APIs for delivering messages along with a set of support routines that process the messages (see Specs: messaging is an interface between the modules, this interface is a set of APIs that provide transport and message services, page 2-5), and is comprised of a message layer which sets up a communication session (see Specs: section 2.1.6, page 2.18), and transport layer which defines how information will be shared (Heil coordinate retrieval of shared information, col 10/lines 66-col 11/line 11).

Regarding claim 7, as discussed on claims 1/14, further a communication layer which defines a mechanism for communications between the system driver layer and the device driver layer (see Specs: I_2O communication layer (messaging service layer), which delivers I/O transactions messages from one software module to another, any where in the I_2O domain, independent of the modules hierarchy, section 2.1.3, page 2-4, Fig. 2-3).

Regarding claim 11, wherein said system driver module and said device driver module constitute a single device (Heil, col 10/lines 43-47, stackable device drivers in a single module) that is portable across a plurality of host operating systems and host network platforms (See Specs: to enable device drivers to port across target processors, device driver source code is written in ANSI C, section 1.2, page 1-4), and works interoperable with a plurality of storage devices and host operating systems (See Specs, coexists with existing device drivers, such as device drivers of multiple classes, and device drives can scale across system platforms, from high-end workstations to high-end server, page 1-4, C code provides portability across platforms, section 2.5.2.1, page 1-40).

Regarding claim 12, wherein said system host driver layer and said device driver layer operate in accordance with an I_20 specification for allowing storage devices to operate independently from the operating system of the host server (Heil, 10, lines 28-42, computer nodes of a server, see abstract).

Regarding claim 13, this claim is substantially the same to claim 2, as discussed, same rationale is applicable.

Regarding claim 17-18, this claims is substantially the same as claims 11-12 as discussed, same rationale is applicable.

5. Claims 19-28 are rejected under 35 U.S.C. 103(a) as being unpatentable over Heil et. al. (Heil) U.S. Patent No. 6,173,374 in view of Intelligent I/O (I2O) Architecture Specification (Specs), version 1.5, March 1997.

Regarding claim 19, wherein, upon initialization, a software 250 driver (Local Transport) scans the local bus so as to locate and initialize all local input/output platforms (IOPs) and builds an opaque "context" structure for each input/output platform (IOP);

Specs: teach building a list of designations for each IOP, wherein after the IOP loads, it executes its initialization sequence, causing the IOP to scan its physical adapters, load and initialize appropriate devices, and build a logical configuration table, section 2.1.7.2, page 2.25, located IOP is added to the system configuration table, host then gives each IOP a list of all IOPs,

section 2.1.4.1, page 2-11, said table lists all the IOP's registered devices and their availability, item 8, page 4-65, for each device registered, the driver provides a list of message handlers, one for each message function, received messages handlers are used to builds a message dispatch table, section 2.2.4.3, page 2-32, each registered I2O device provides message handlers that process the messages to the device, the identify of the event handler is accompanied by a context variable so that the message variable can determine the associated I/O transaction of each registered IOP, section 2.2.4.1, section Page 2-32.

wherein software driver 240 (Remote Transport) prepares to accept requests from a remote server through said computer network, (Heil: Software drivers 240 arranged to manage the reception of remotely generated requests from the network for local data to be exported onto the computer network, col 11/lines 36-46, remote computer nodes of a server cluster, abstract), and

means for building an IOP descriptor structure for each input/output platform (IOP) which includes an exported table of function call pointers and the context required by the software driver 250 to communicate with the input/output platform (IOP);

Specs: Each I_2O device is created with a unique TID and associated with an event queue, IOP and driver communicate via API function calls. Driver supplies a pointer to the (IOP descriptor function table) message dispatch table when the device is created. This table lists Function codes, their priorities and message handlers for processing request messages. When communicating with the IOP, the IOP receives a request message to that TID, the IOP uses the message's Function code (function call), queues the request to the event queue. If the driver sends requests, it must register their reply handlers (i.e. send/receive handlers) and priorities via an API function call. Handles are placed in a structure and which returns an InitiatorContext value identifying that structure. When a Driver sends a request, it uses the appropriate InitiatorContext value. When a reply is received, the IRTOS retrieves the reply handler and priority from a structure identified by the InitiatorContext field and then queues the event, section 5.1.5, page 5-4, section 5.2.6, page 5-14; In this manner an IOP descriptor structure for each input/output platform (IOP) is built, which includes an exported table of function call pointers and the context required by the driver to communicate with the input/output platform (IOP); connection using said direct call path between the driver 250 and driver 240, discussed above in claims 1 and 14;

and means for establishing a network management communication channel through the driver 240 (Heil, col 11/lines 54-65, shipping via 240, col 10/line 65-col 10/line 17), and means

which waits for a connection (see Specs, table 4-5, page 4-16, connection_fail, timeout waiting for response), however neither Heil nor Specs teach determining the number of IOP;

Bonola teaches means for determining upon initialization and a starting process, the number of CPU's present in the computer system along with the context of the computer system, col 9/lines 55-58, teaching means for querying so as to determine the number of input/output processing (IOP), col 8/line 5-13, 58-64);

It would have been obvious to one ordinary skilled in the art at the time the invention was made to modify existing teachings with means for determining the number of IOP as taught by Bonola motivation would be to prevent error in the installation phase determining the actual of detected processor, utilizing said number and the context information to initialize the drivers, supporting the allocation of shared memory without requiring extra memory, as in the prior art.

Regarding claim 20, wherein said software driver 250 further has a send handler function and said driver 240 further has a receive handler function which are respective program interfaces for receiving an inbound message from a remote server on said computer network for direct access to local IOP and for delivering an outbound message to said remote server on said computer network.

Specs teaches wherein for IOP- IOP communication, inbound/outbound message queue location are exchanges, section 2.1.4.1, and page 2-11; when communicating with the IOP, the IOP receives a request message to that TID, the IOP uses the message's Function code (function call), queues the request to the event queue. If the driver sends requests, it must register their reply handlers via an API function call. When a Driver sends a request, it uses the appropriate InitiatorContext value from handles are placed in a structure and which returns an InitiatorContext value identifying that structure. When a reply is received, the reply handler are retrieved from a structure identified by the InitiatorContext field and then queues the event, section 5.1.5, page 5-4, section 5.2.6, page 5-14; In this manner an IOP descriptor structure for each input/output platform (IOP) is built, which includes an exported table of function call pointers and the context required by the driver to communicate with the input/output platform (IOP); In this manner send/receive (request/reply) handlers are exchange for corresponding inbound and outbound messages between the local and remote IOP modules.

Regarding claim 21, wherein said Remote Transport further builds an IOP connection structure including at least an IOP descriptor pointer which refers to the IOP descriptor structure of the Connection Manager for making a direct call to the Local Transport through the receive handler function and the send handler function.

Specs teaches a connection message structure is used by an IOP to connect one of its drivers and a device registered on another IOP, IOPs using send/receive request messages, messages comprising descriptor structure (e.g. InitiatorDevice and TargetDevice) used to refer to the driver and IOP that will send requests and the device on IOP2 that will receive them, section, page 4-20, said descriptor identifies the driver and the IOP, adapters and device types, section 5.2.1, page 5-7; IOP descriptor structure for each input/output platform (IOP) is built, which includes an exported table of function call (send/receive handler) pointers and the context required by the driver to communicate with the input/output platform (IOP); connection using said direct call path between the driver 250 and driver 240, discussed above in claims 1 and 14; wherein said connection request and its reply convey information enabling the two IOPs to establish a direct path for exchanging messages, section 2.1.4.1, and page 2-11.

Regarding claim 22, as discussed on claims 1 and 14, further providing a direct call access to said storage devices for said remote server to share resources of said storage devices by exporting said resources, as discussed above, further while bypassing operating system protocol stacks; Heil: software driver 240 and 250 discussed above, provide exporting functions will bypassing operating system protocol stacks 260, col 10/line 66-col 35 enabling exporting supported by software driver 250 comprising a cache maintaining updated data from the data from the local blocks 118, this supports the bypassing of protocol stack 260, for exporting local storage device onto a computer network, where I/O shipping 270 managing the reception of remotely generated requests from the other system network for local data to be exported onto the computer network, and I/O shipping HDM 280 manages the communication medium, as discuss above.

Regarding claim 23, the claim comprise a combination of previously discussed claim limitation, these claims are 22, 1, 7, and 19.

Regarding claim 24, this claim comprises the same limitations as those in claim 4, wherein Local Transport is not denoted system driver module

Regarding claim 25, this claim comprises the same limitation as those in claims 5 and 6, same rationale is applicable.

Regarding claim 26, this claim comprised the same limitations as in claim 11 and 12,same rationale is applicable.

Regarding claims 28-27, this claims are substantially the same as claims 19-20, respectively, same rationale is applicable.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to **Prieto, B.** whose telephone number **is (703) 305-0750**. The Examiner can normally be reached on Monday-Friday from 6:30 to 4:00 p.m. If attempts to reach the examiner by telephone are unsuccessful, the Examiner's Supervisor, **Mark H. Rinehart** can be reached on **(703) 305-4815**. The fax phone number for the organization where this application or proceeding is assigned is **(703) 308-6606**. Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist whose telephone number is **(703) 305-3800/4700.** Any response to this action should be mailed to:

Commissioner of Patents and Trademarks
Washington, D.C. 20231

or faxed to:

(703) 746-7239, (for Official communications intended for entry)

Or:

(703) 746-7240 (for Non-Official or draft communications, please label "PROPOSED" or "DRAFT")

Hand-delivered responses should be brought to Crystal Park II, 2121 Crystal Drive, Arlington. VA., Fourth Floor (Receptionist), further ensuring that a receipt is provided stamped "TC 2100".

B. Prieto
Patent Examiner
January 28, 2002

LE HIEN LUU
PRIMARY EXAMINER